Lab 3: Digital Modulation & Transmission - II

EEP55C26 Open Reconfigurable Networks
Lingyu Gong

I. Lab purpose

In this lab, we aim to understand coherent digital modulation in wireless communication by focusing on Carrier Synchronization and Symbol Timing Recovery. Using real-world signals and Software Defined Radios (SDRs), the experiment highlights the importance of aligning the receive carrier with the transmit carrier in frequency and phase, and precisely determining symbol timing for accurate message recovery. This practical approach emphasizes the critical roles these processes play in coherent communication systems, offering insights beyond traditional hardware methods.

II. Carrier Synchronisation and Symbol Timing Recovery in BPSK

1. Creating the Simulink model

This whole structure basically follows this workflow: Receive the signal from transmitter, setting the fixed offset value to estimate the offset, add gain module to make sure the signal receives enough power to estimate error protect original signal. After receive the signal, this signal will pass AGC module, which is an automatic gain control system, this block will adjust the gain adaptively and make sure the output amplitude will keep constant. After this signal will go through Carrier Synchronizer, which will adjust the carrier frequency and phase offset by compensate the offset after coarse calibration of the carrier frequencies. Then go to square root that can raise cosine filter for pulse shaping in the received signal path. After, the signal will pass Symbol Synchronizer block adjust the sampling rate to ensure the incoming pulses are sampled at the "opening" of the eye. Final we have BPSK block, which is the binary phase shift keying, a digital modulation technique used to encode data on a given carrier waveform.

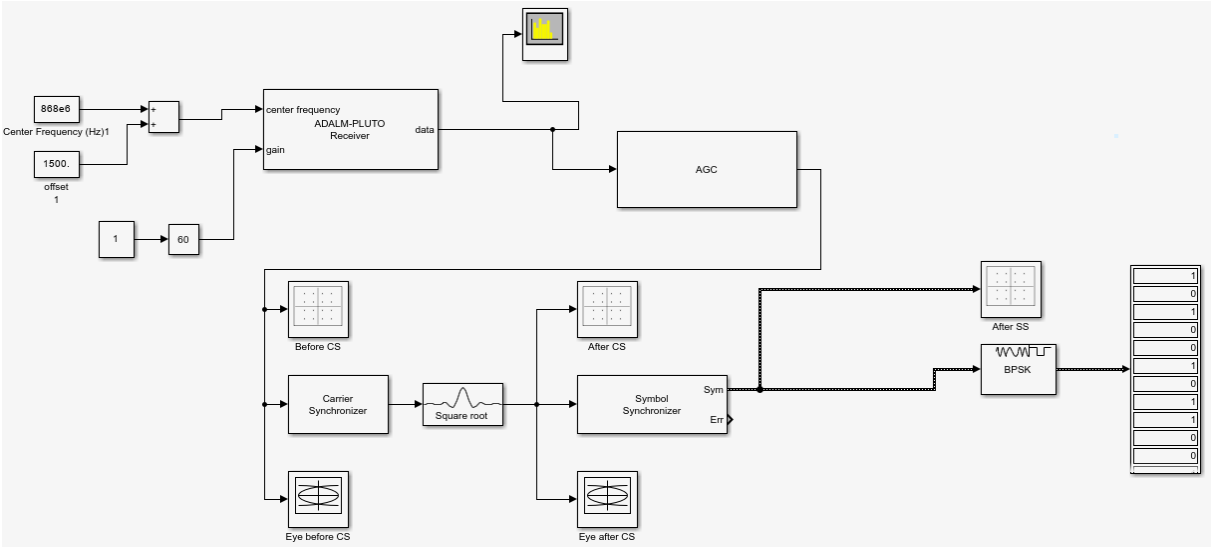

Figure 1-1 Simulink model for a BPSK receiver using PLUTO.

Question 1.1: Observe the Eye Diagram plots at the input and output of the Carrier Synchronizer block. Using a screenshot of the plots, explain any observations and/or interpretations.

We can see that before Carrier Synchronisation, the eye diagram is basically unrecognisable and the noise and interference is very strong, but after Carrier Synchronisation, it is clear that the eye diagram is "open" and the interference is reduced. This is mainly due to the fact that the module can reduce the frequency and phase shifts in order to reduce the noise and interference of the signal.

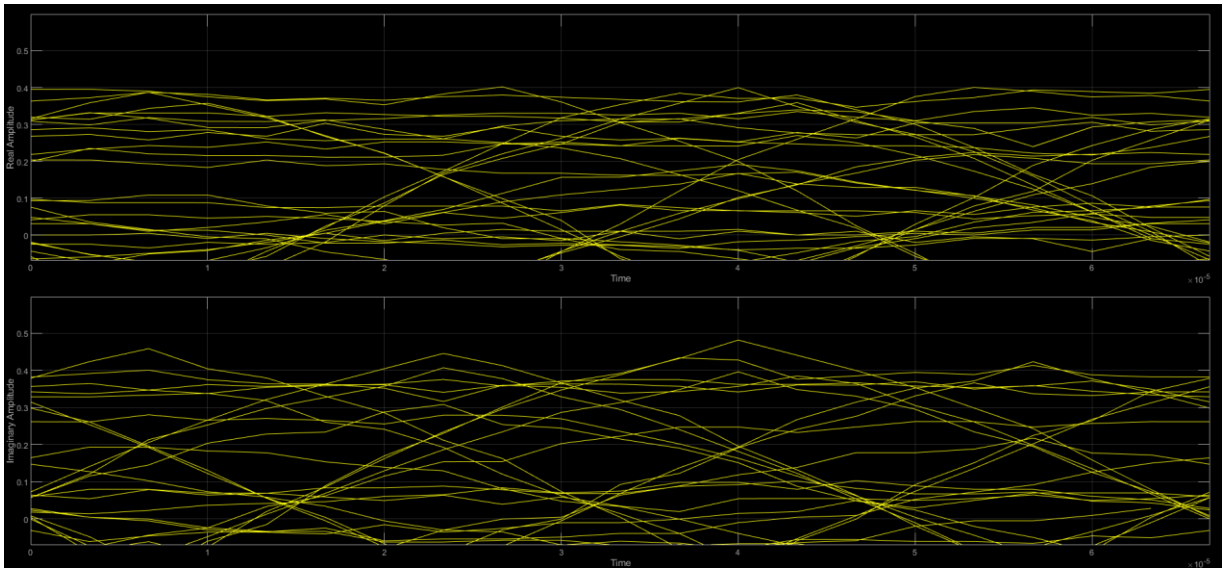

Figure 1-1 Eye Diagram before Carrier Synchronisation block.

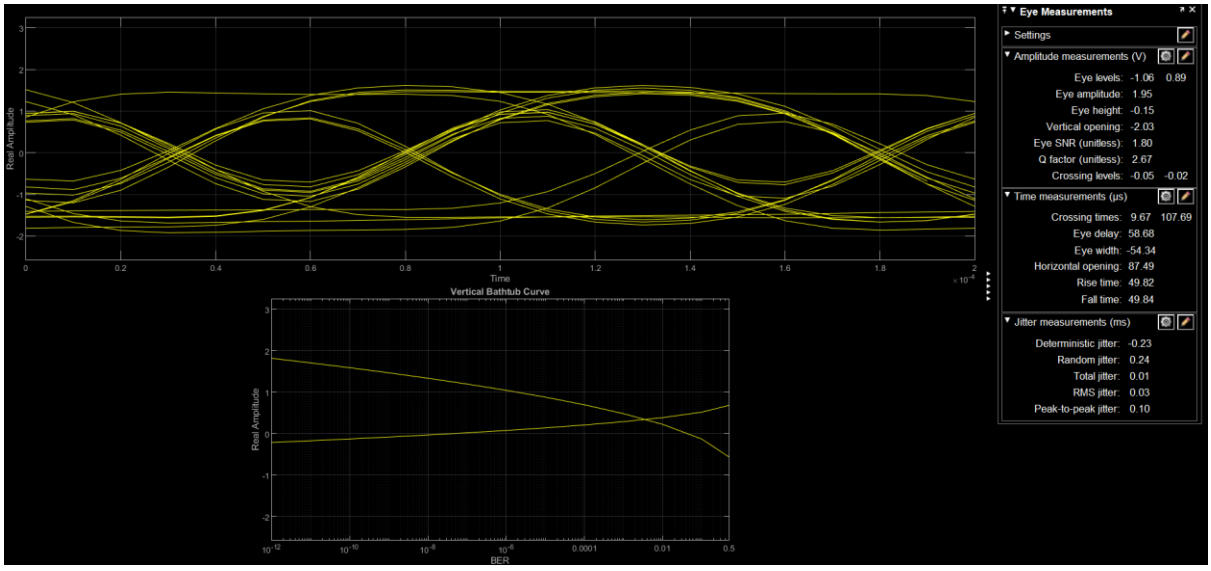

Figure 1-2 Eye Diagram after Carrier Synchronisation block.

Question 1.2: Observe the Constellation Diagram plots at the input and output of the Carrier Synchronizer block. Using a screenshot of the plots, explain any observations and/or interpretations.

As we can see in the first constellation graph, the receive signal were in a circle due to the offset from phase aspect, because with the "t" changes, these detected points will rotate in some angle, and become like a circle int the end. However, in the right graph, after the carrier Synchronizer, phase offset was fixed by the adjustment in this block, so it returns back to correct place.

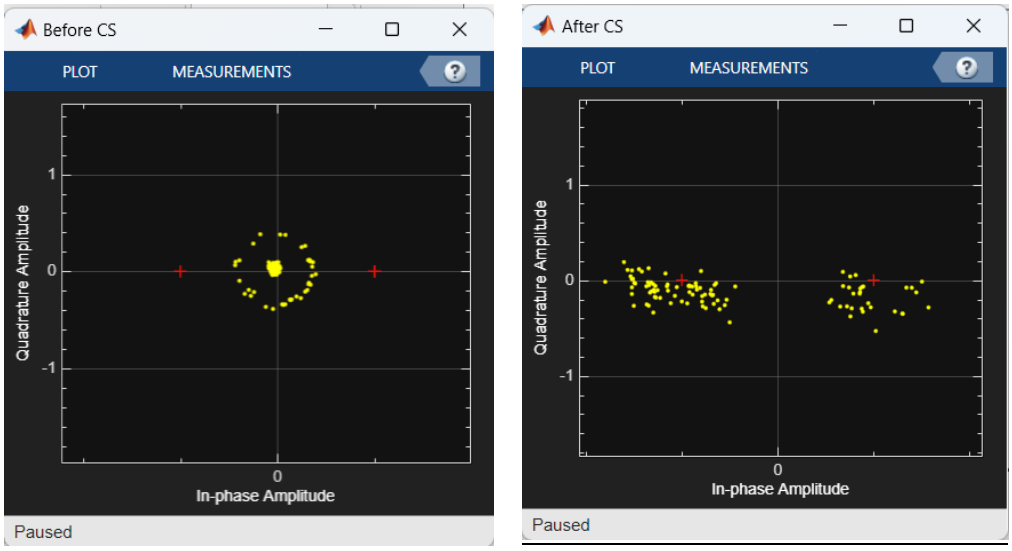

Figure 1-3 Constellation chart observations before and after Carrier Synchronizer.

Question 1.3: Observe the Constellation Diagram plots at the input and output of the Symbol Synchronizer block. Using a screenshot of the plots, explain any observations and/or interpretations.

It can be clearly seen that before the Symbol Synchronizer the received signal point distribution is not at the target position, but slightly shifted sideways, but after the Symbol Synchronizer it returns to the expected position. This is mainly due to the fact that the sampling rate is adjusted in the module so that the sampling positions are at the expected positions that can be observed.

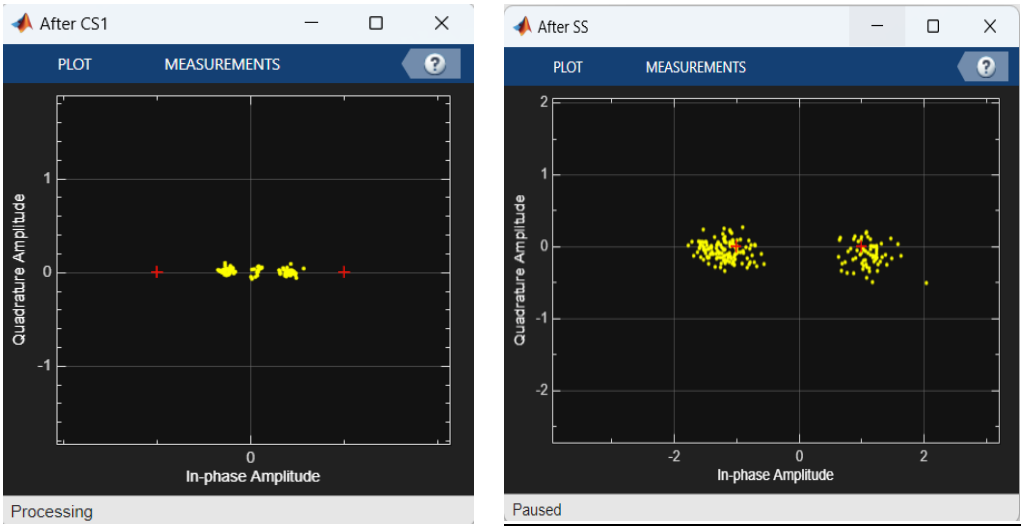

Figure 1-4 Observation of constellation charts before and after Symbol Synchronizer.

Question 1.4: Observe the effects of reduced signal-to-noise ratio (SNR) at the input to the receiver by • decreasing the Gain of the PLUTO-SDR Receiver • placing an object near/over the receive antenna Examine the effects of reduced SNR on Constellation Diagram and Eye Diagram plots. Take screen capture of your observations and provide any inferences you may have.

For my experiment, with the same structure as before, I tested the Gain parameter from 5 to 70, with an interval difference of 5. I found that the results were similar around Gain =30, and at around Gain =50, the effect was pretty good. However, at Gain =70, it seems like a different result pattern. Therefore, in my report, I mainly focused on the differences between the three Gain settings and provided a simple analysis.

The first three graph shows before Carrier Synchronizer, from left to right they were Gain=30 Gain=50 Gain=70. We can see there are more noise due to the offset on the signal, and when in Gain=70 the noise separate into two circle.

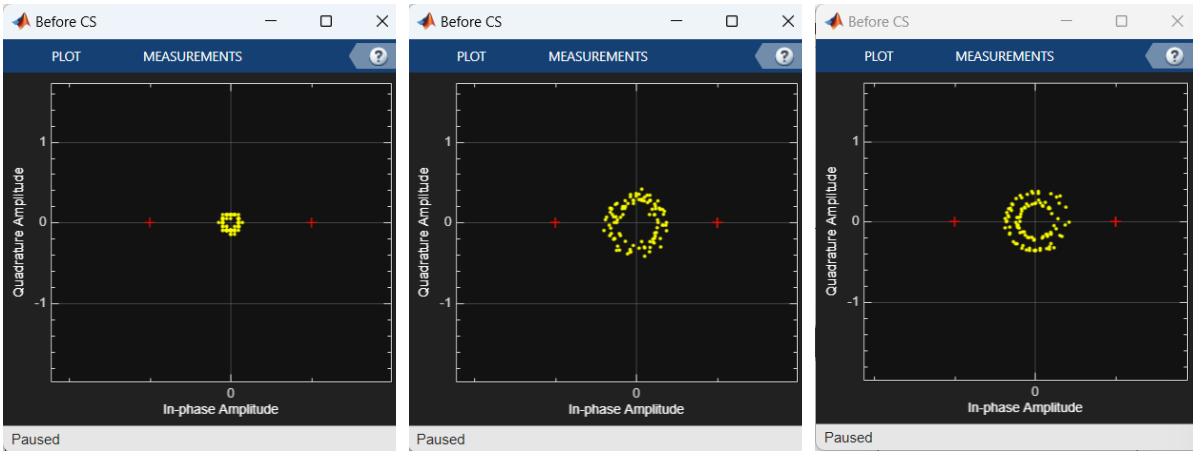

Figure 1-5 Constellation observations before Carrier Synchronizer (Gain=30; Gain=50; Gain=70).

The second parts these graph shows the observation for three different Gain values on the Eye Diagram before and after Carrier Synchronizer, still using Gain=30 Gain=50 Gain=70.

What is clear is that it is difficult to see a complete and clean Eye Diagram until it has been subjected to CS, whereas the location and characteristics of the Eye Diagram eyes can be seen more clearly after processing with CS.

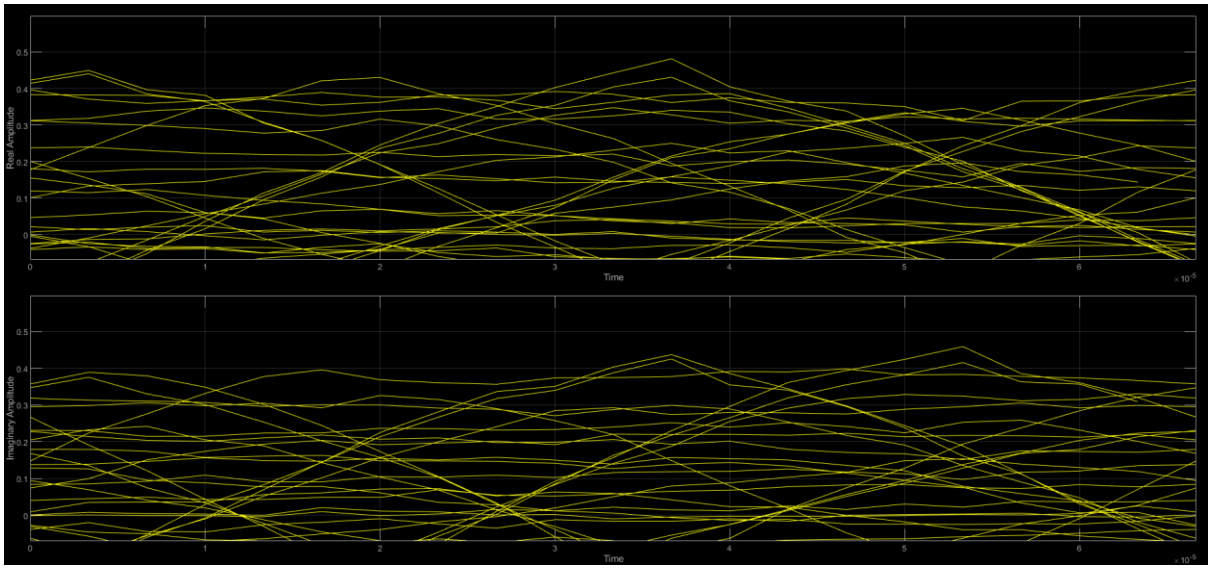

Figure 1-6-1 Eye Diagram before Carrier Synchronizer (Gain=30).

When Gain=30, the shape of the eyes can be seen, but they are barely open.

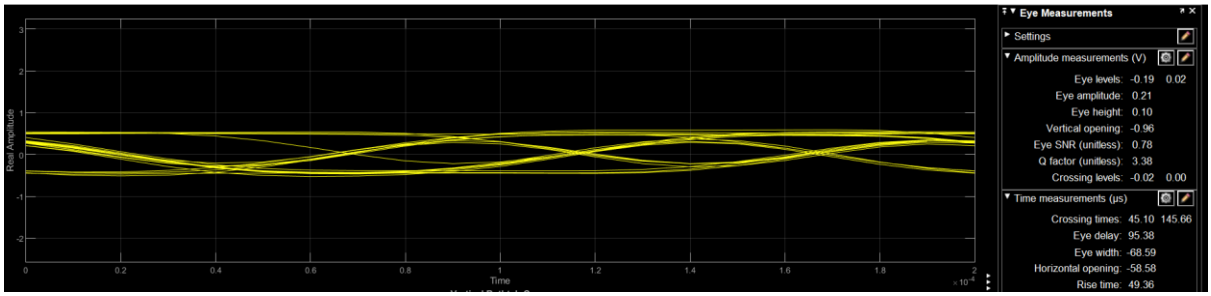

Figure 1-6-2 Eye Diagram After Carrier Synchronizer (Gain=30).

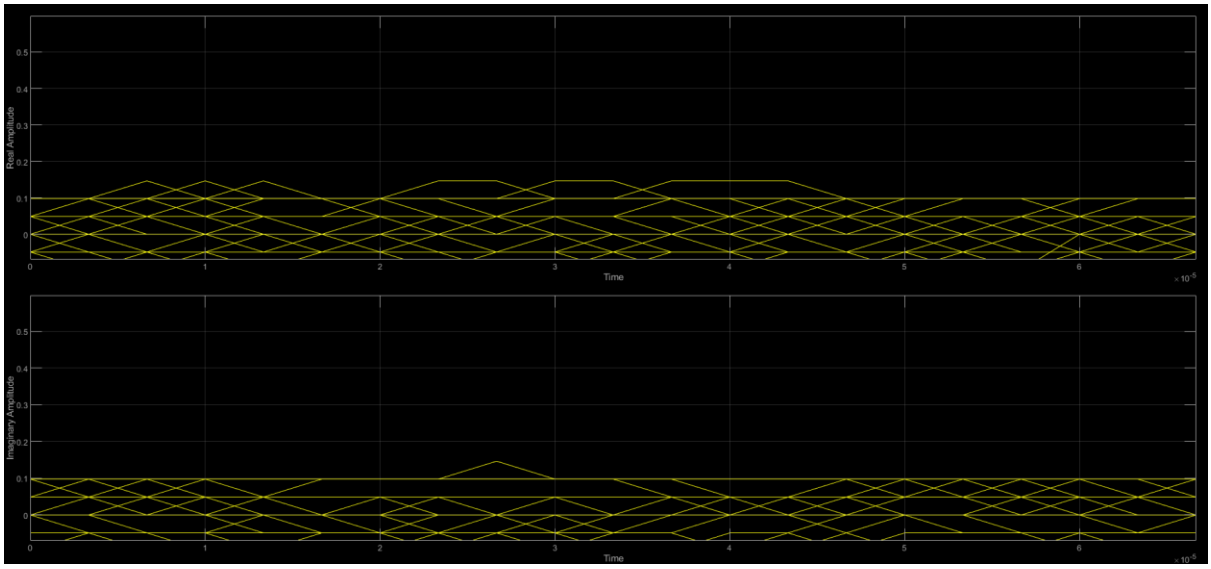

Figure 1-7-1 Eye Diagram before Carrier Synchronizer (Gain=50).

When Gain=50, the shape of the eyes can be clearly seen, and the eyes open more widely.

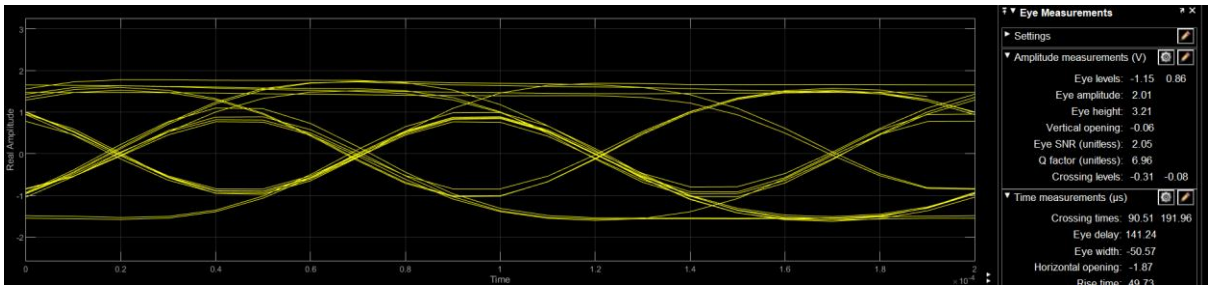

Figure 1-7-2 Eye Diagram After Carrier Synchronizer (Gain=50).

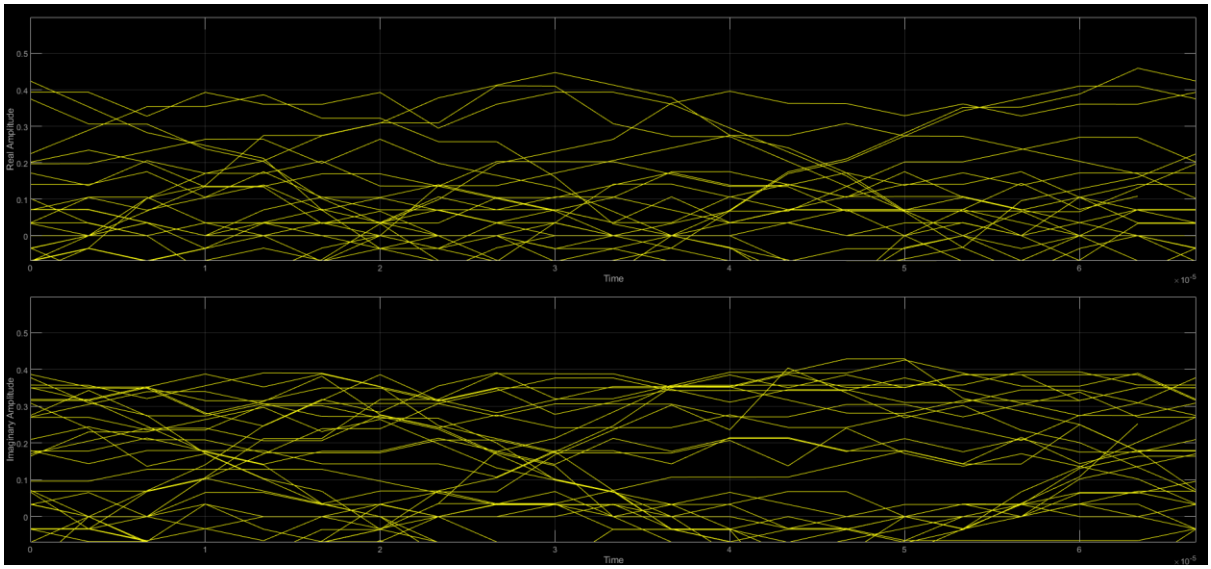

Figure 1-8-1 Eye Diagram before Carrier Synchronizer (Gain=70).

When Gain=70, the shape of the eyes can be seen more clearly, with a lot of noise around them, suggesting that Gain=70 increased the Power of the noise too much causing it to become noticeable, and that the eyes opened less, but better than Gain=30.

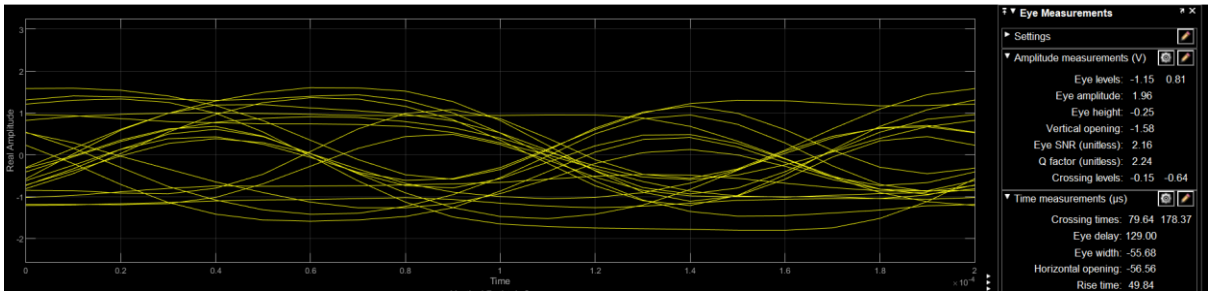

Figure 1-8-2 Eye Diagram after Carrier Synchronizer (Gain=70).

The next step will be to compare the changes in the constellation diagrams before and after SS, with the same constraints as in the previous case, and the comparison diagrams are, from left to right, Gain=30 Gain=50 and Gain=70.

Similarly to the previous one, we can see that before going through SS, the lack of reprogramming for the sampling points leads to a large offset, which is far from the expectation, but after the effect of SS, it is significantly improved.

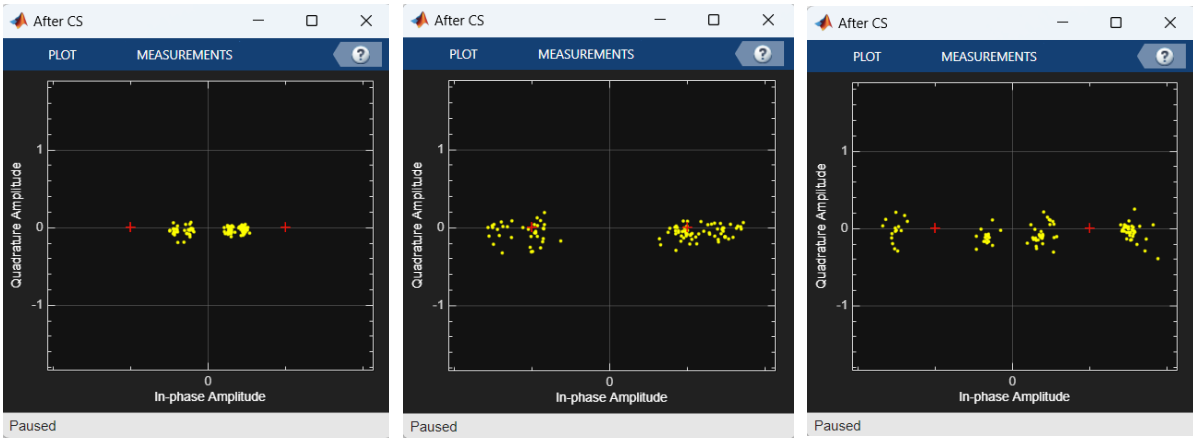

Figure 1-9 Constellation after Carrier Synchronizer.

It can be seen that the gain provided at Gain = 30 is too small, resulting in an inability to remove the effects of offset and noise, which is significantly improved by increasing it to 50, and there is no significant change by increasing it to 70.

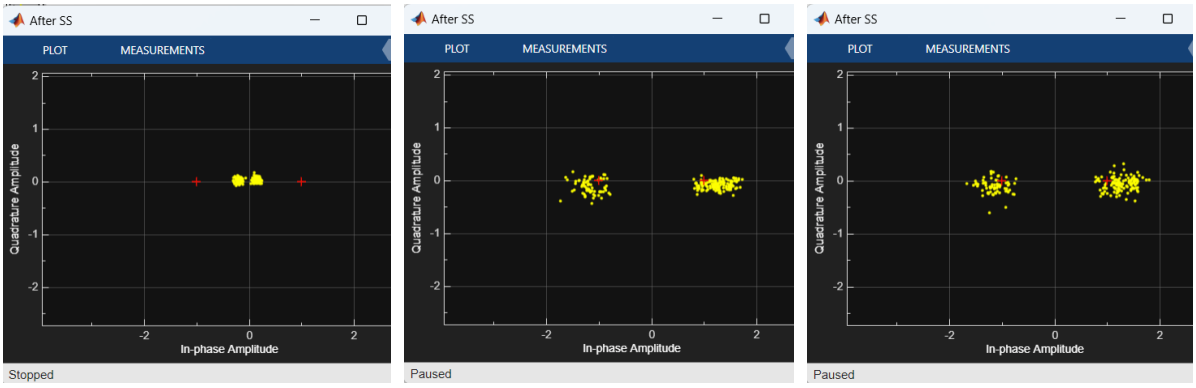

Figure 1-10 Constellation after Symbol Synchronizer.

If I manually block the receive antenna, I get the following. It can be seen that the received signal receives a physical blockage, resulting in less energy and more noise, bringing unstable results.

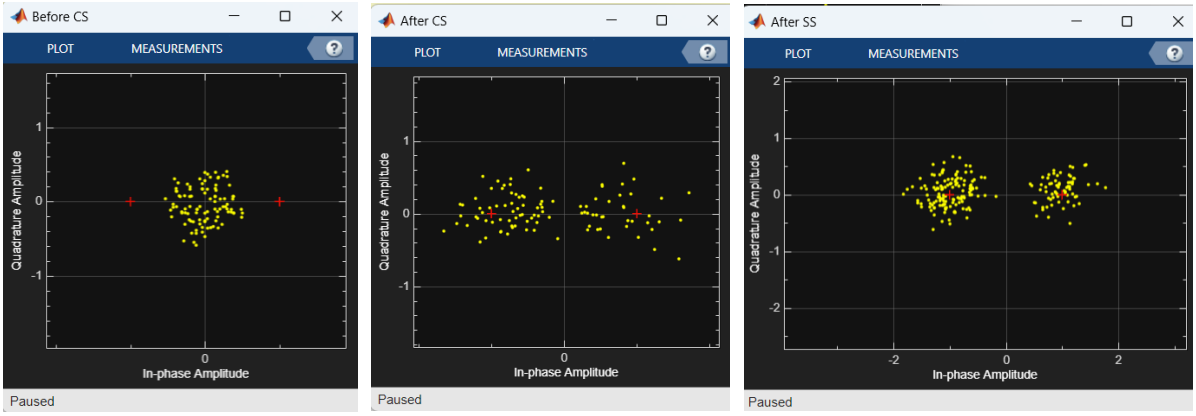

Figure 1-11 Constellation after block the receive antenna.

It can be seen that there is a lot of interference in the eye diagram because of the physical blocking noise becoming larger.

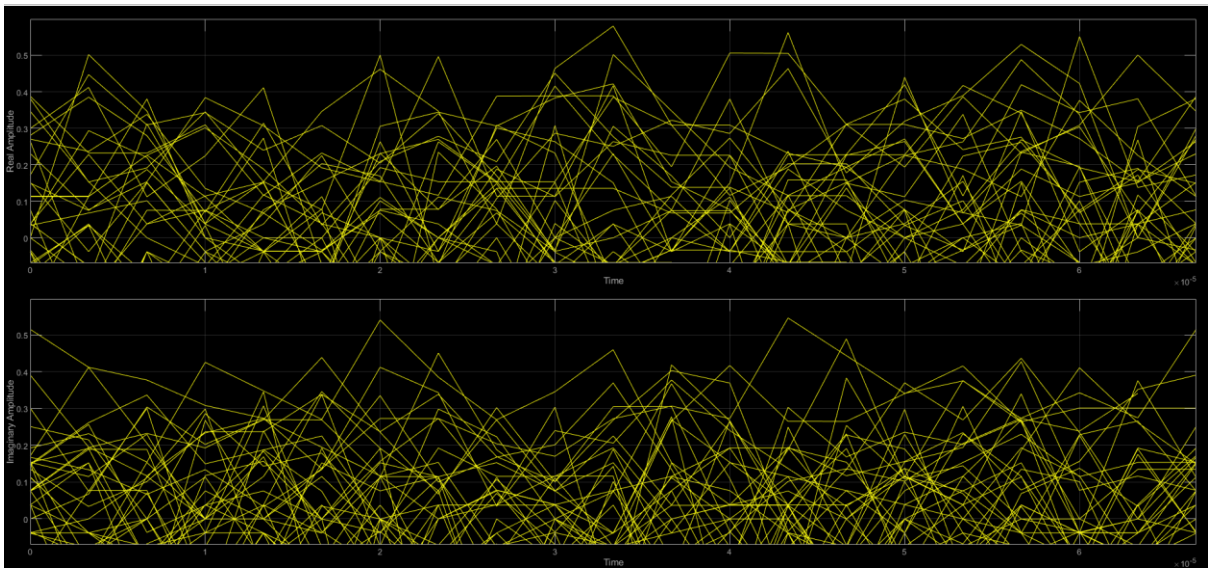

Figure 1-12 Eye Diagram before SS (block the receive antenna).

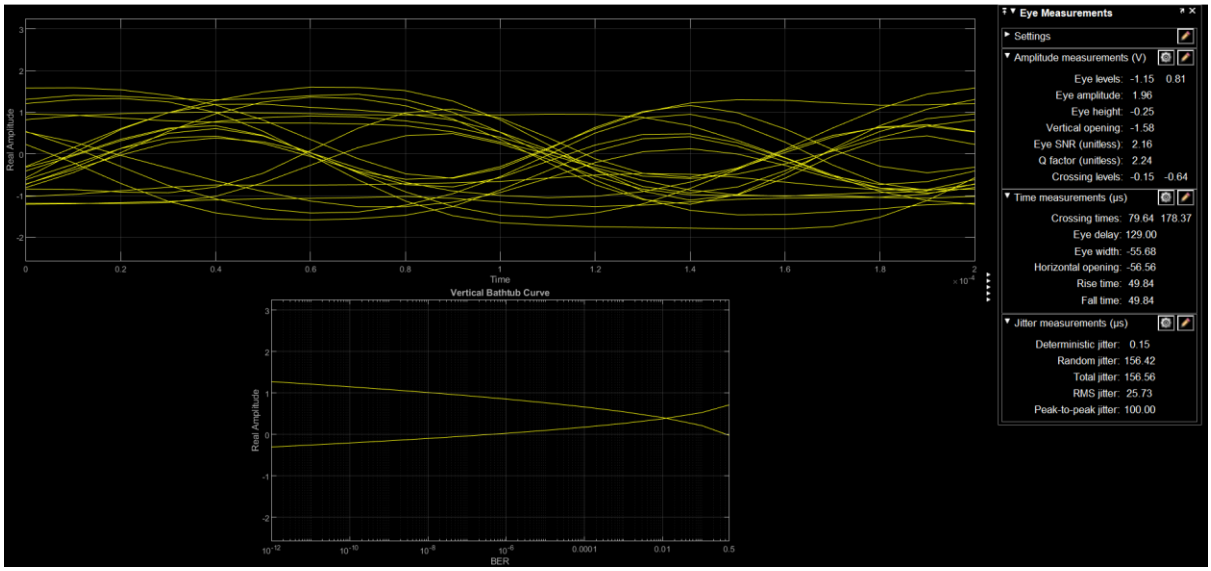

Figure 1-13 Eye Diagram after SS (block the receive antenna).

2. Differential BPSK

2.1 D-BPSK communication

Figure 2-1 shows the Simulink model for a D-BPSK communication.

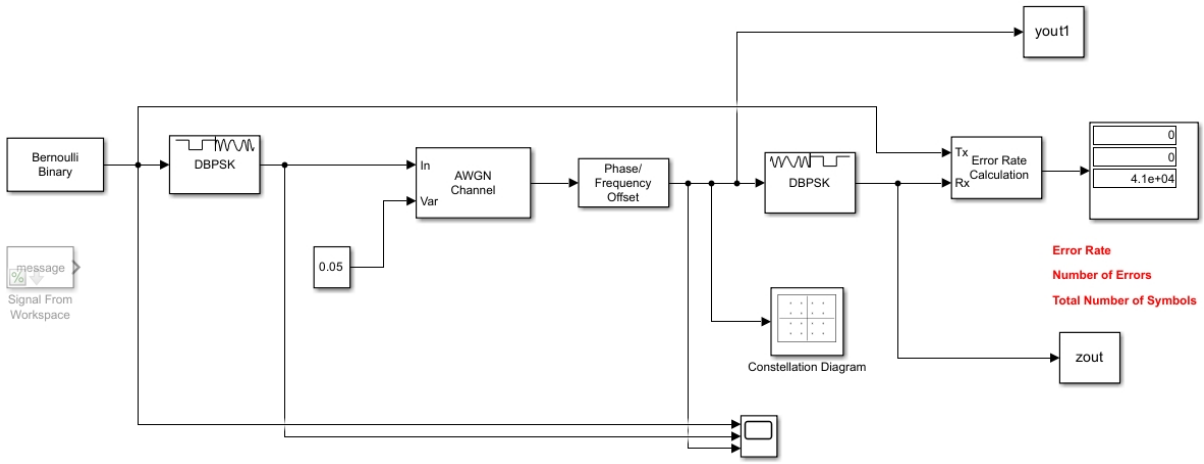

Figure 2-1 Simulink model for a D-BPSK communication

Result shows below: From the wiring of the time domain plot, it can be seen that the first part (red boxed area) is the unmodulated signal, the second part (blue boxed area) has been modulated with DBPSK and the conversion of zeros and ones can be seen, and the third part (yellow boxed area) has been subjected to a module with added noise, and it can be seen that the smooth line becomes zigzagged, implying an increase in noise.

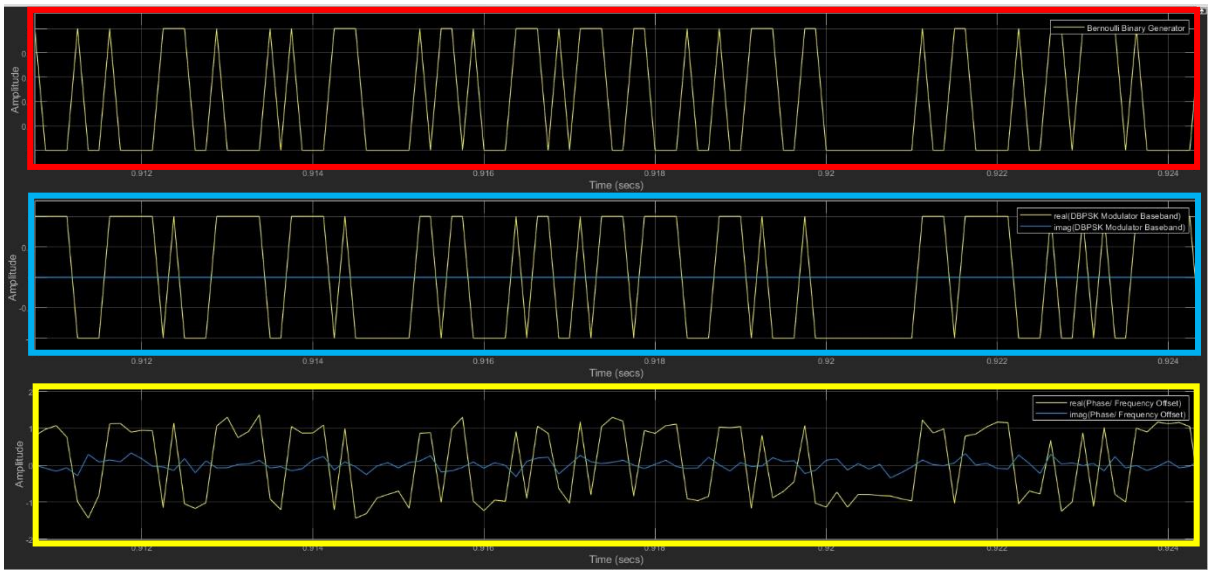

Figure 2-2 Time Domain Comparison.

Phase/Frequency Offset: This module is used to apply frequency offsets and phase offsets to the received signal. On a constellation diagram, these offsets will appear as a rotation of the symbol dots (phase offset) and/or a movement along the circle (frequency offset).

If I change the phase offset to 50, the resulting graph appears on the left. However, if I change the frequency offset to 20, the graph appears as shown on the right. It appears that changing the phase offset only rotates the points on the graph, while changing the frequency offset alters the distribution of the signal and makes it looser, similar to a traffic circle.

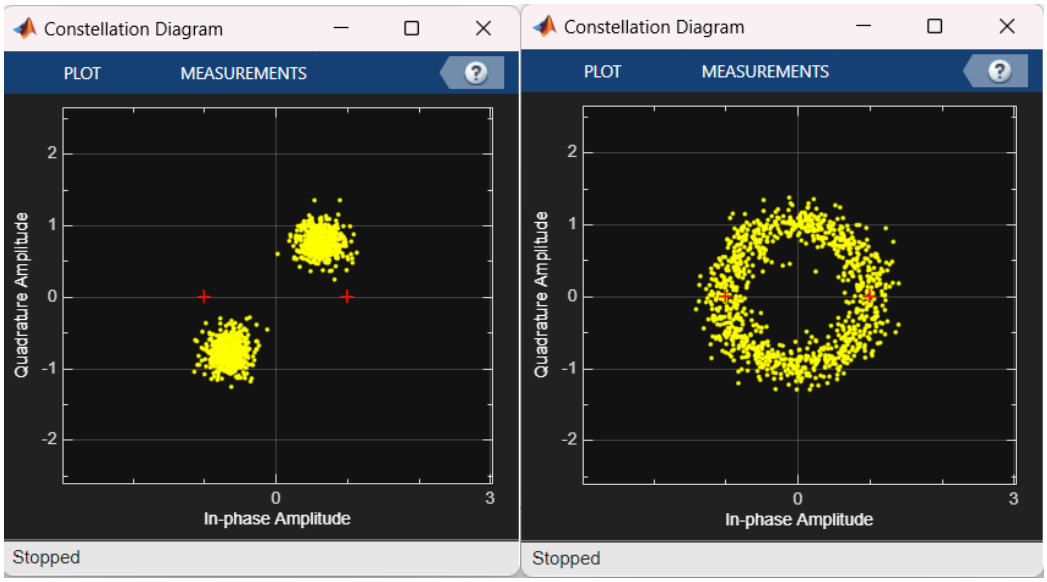

Figure 2-3 Phase/Frequency Offset Analyzation.

2.2 D-BPSK Demodulator

Question 2.1: Differential encoding is quite useful to overcome the phase noise issues that you encountered with standard BPSK. However, what could be a potential issue with building them using the Pluto devices for our lab?

Potential problems with the use of differential coding may include device synchronisation issues. In differential coding, the meaning of each symbol depends on the previous symbol. If the

synchronisation mechanism is inaccurate, this can lead to a cascade of incorrect decoding, as one error can affect the decoding of all subsequent symbols. For Pluto devices, clock desynchronisation may increase this risk, especially when signals are transmitted over radio channels where clock drift is more common due to multipath effects or other channel influences.

Question 2.2: Describe in a few lines your method for decoding the differentially encoded symbols. Include the salient parts of your MATLAB code you wrote to perform the decoding.

This code retrieves the signal data from yout.message_rx.signals.values, strips off the guard bands at the beginning and end, and converts the complex signal to phase information. The script then calculates phase differentials, wrapping them within the [0, 2π] range. These phase differences are used to decode the binary data, assuming a phase change greater than π/2 indicates a bit flip. Finally, the binary data is converted into ASCII characters and displayed, revealing the transmitted message.

```
% Compute the phase of the data
phases = angle(data);

% Compute the phase differentials and wrap them to the range [0, 2*PI]
phaseDiffs = diff([phases(1); phases]);
phaseDiffsWrapped = wrapTo2Pi(phaseDiffs);

% Decoding based on phase differentials
decodedBits = phaseDiffsWrapped > pi/2;

% Reshape the data for ASCII conversion
numChars = floor(length(decodedBits) / 8);
decodedChars = reshape(decodedBits(1:numChars*8), 8, numChars).';

% Convert binary to ASCII characters
asciiVals = bi2de(decodedChars, 'left-msb');
decodedMessage = char(asciiVals).';
```

Figure 2-4 Main Code.

Question 2.3: What is the text message?

P2Good Afternoon, I am your Pluto Board no:10 Good luck with part-2 of 4C21!\n